Numer. Math. Theor. Meth. Appl.
doi: 10.4208/nmtma.2014.1304si

Vol. **7**, No. 4, pp. 473-498
November 2014

A Projection Preconditioner for Solving the Implicit Immersed Boundary Equations

Qinghai Zhang[1,*], Robert D. Guy[2] and Bobby Philip[3]

[1] *Department of Mathematics, University of Utah, Salt Lake City, UT 84112, USA.*
[2] *Department of Mathematics, University of California Davis, Davis, CA 95616, USA.*
[3] *Oak Ridge National Laboratory, Oak Ridge, TN 37831, USA.*

Received 17 April 2013; Accepted 15 April 2014

Available online 11 November 2014

Abstract. This paper presents a method for solving the linear semi-implicit immersed boundary equations which avoids the severe time step restriction presented by explicit-time methods. The Lagrangian variables are eliminated via a Schur complement to form a purely Eulerian saddle point system, which is preconditioned by a projection operator and then solved by a Krylov subspace method. From the viewpoint of projection methods, we derive an ideal preconditioner for the saddle point problem and compare the efficiency of a number of simpler preconditioners that approximate this perfect one. For low Reynolds number and high stiffness, one particular projection preconditioner yields an efficiency improvement of the explicit IB method by a factor around thirty. Substantial speed-ups over explicit-time method are achieved for Reynolds number below 100. This speedup increases as the Eulerian grid size and/or the Reynolds number are further reduced.

AMS subject classifications: 65M55, 65F08, 76M20, 76D99

Key words: Fluid-structure interaction, immersed boundary method, projection method, preconditioning.

1. Introduction

The Immersed Boundary (IB) method introduced by Peskin [19, 20] has been a popular approach for simulating fluid-structure interactions. Physical variables for the fluid are discretized on an Eulerian grid while those for the immersed boundary are discretized on a Lagrangian grid. The fluid satisfies the no-slip condition on the immersed

*Corresponding author. *Email addresses:* qinghai@math.utah.edu (Q. Zhang), guy@math.ucdavis.edu (R. D. Guy), philipb@ornl.gov (B. Philip)

boundary, which means that the Lagrangian grid points move at a velocity interpolated from the Eulerian grid. Deformations of the immersed boundary generate elastic forces which are transmitted to the fluid through a forcing term added to the governing equations of fluid dynamics. In this manner the IB method provides much flexibility in modeling the coupling between the Eulerian and Lagrangian variables, since explicitly enforcing boundary conditions at the fluid-structure interface is avoided.

The popularity of the IB method is partly due to its simplicity. In a typical explicit-time method, the Eulerian velocity and pressure fields are updated for a fixed configuration of the immersed structure, and then the position of the Lagrangian structure is updated from the newly computed velocity field. This approach effectively decouples the Eulerian and Lagrangian equations, and solvers are needed only for the Eulerian equations (i.e., the incompressible Stokes or Navier-Stokes equations), for which fast Cartesian grid solution methods are available. The implementation is straightforward since it only entails augmenting one's favorite fluid solver with the IB forcing term. Nonetheless, when an elastic boundary becomes stiff, explicit-time IB methods suffer from either instability or restrictively small time steps.

To remedy the severe time step restriction of explicit IB methods, a number of implicit and semi-implicit schemes have been developed. However, their implementation is much more involved and is a subject of ongoing research; see for example [3, 4, 11, 13, 15–18, 22, 24] and references therein. These methods are centered at answering two essential questions:

(A) How does stiffness affect the stability of the numerical solver?

(B) How to efficiently solve the discretized equations that are highly stiff?

Clearly these two questions are closely related. It had been commonly believed that only fully implicit discretizations could produce an unconditionally stable IB method until the work of Newren *et. al.* [17]. They showed that semi-implicit versions of backward Euler and Crank-Nicolson schemes can be made stable so long as the spreading and interpolation operators are evaluated at the same time instant and the same spatial location. When this is satisfied, the total energy of the numerical system does not increase over time even if the evaluation of the spreading and interpolation operators are lagged in time. This conclusion not only answers question (A), but also partially answers question (B), since the lagged evaluation of the spreading and interpolation operators opens up exciting possibilities of unconditionally stable discretizations via *linear* systems.

Many implicit methods use a Schur complement approach to reduce the coupled Eulerian-Lagrangian equations to purely Lagrangian equations [3, 4, 16]. These methods achieve a substantial speed-up over explicit methods when there are relatively few Lagrangian mesh nodes. In addition, some methods require that the boundaries be smooth, closed curves [13]. An open question is whether there exist robust, general-purpose implicit methods that are more efficient than explicit methods, or whether specialized methods must be developed for specific problems.

Newren et al. [18] studied the efficiency of a group of semi-implicit linear solvers based on double Schur complement. Depending on the parameters of the tests, their semi-implicit linear solvers were between two times slower and two times faster than the explicit IB method. However, those results were obtained without preconditioning. With appropriate preconditioning, we expect that this approach will offer a significant improvement over explicit methods. In [11], we developed a geometric method for a model of the implicit IB equations which included the viscous terms and immersed boundary terms, but it ignored the pressure and the incompressibility constraint. This multigrid method was an excellent preconditioner for the model problem.

This paper addresses question (B) as a followup of our previous work in [11,17,18]. By eliminating the IB variables via a Schur complement, we formulate the IB equations as a saddle point problem of the fluid variables. It is well-known that the efficiency of solving a saddle point problem depends largely on preconditioning. As pointed out by Benzi, Golub, and Liesen [2], there does not exist a "best" preconditioner for saddle point problems in a general sense, and frequently efficient preconditioners are designed by accounting for the physics of the specific problem. Indeed, general preconditioners based on purely algebraic techniques such as the incomplete LU factorization are found to perform poorly in the context of incompressible flows.

One possible way to form a preconditioner for the IB saddle point problem is through projection methods which decouple the equations for the velocity and the pressure. This decoupling allows us to use the multigrid method from [11] because the equation to solve the velocity has the same form as that in the model problem for which the algorithm was developed. Although projection methods are not efficient solvers for the IB equations [18], they might lead to excellent preconditioners. It is not a new idea to first precondition a saddle point problem with a projection preconditioner and then solve it by a Krylov subspace method. For example, Griffith [9] has successfully applied a projection preconditioner in solving the single-phase Navier-Stokes equations with general boundary conditions. However, as far as we know, this strategy has not been explored for IB methods.

Utilizing the stable discretization in [17], we derive an ideal[†] preconditioner for the IB saddle point linear system from the viewpoint of projection methods. However, this prefect preconditioner is prohibitively expensive to apply, and so, we perform efficiency studies for a number of preconditioners that approximate this ideal one. Finding an effective projection preconditioner for all regimes of Reynolds numbers proved to be difficult. By truncating a matrix series, we find a projection preconditioner that leads to a substantial efficiency improvement over the explicit IB method for low Reynolds number and high stiffness.

The rest of this paper is organized as follows. In Section 2, we briefly review the IB method, the adverse influence of stiffness on the stability of the explicit IB method, and an implicit discretization scheme that is unconditionally stable. In Section 3, we formulate a generic saddle point framework from the viewpoint of projection meth-

[†]It is ideal in the sense that the preconditioned linear system always converges after two GMRES iterations.

ods, derive the ideal preconditioner, and approximate it by truncating a matrix series. Numerical tests performed in Section 4 show that the proposed method with a simple preconditioner is much more efficient than the explicit IB method in the case of stiff boundaries and large viscosity. Section 5 finally concludes this paper.

2. Analysis

In a bounded domain $\Omega \subset \mathbb{R}^{\mathrm{D}}$, we numerically solve the incompressible Navier-Stokes equations with an immersed boundary Γ:

$$\nabla \cdot \mathbf{u} = 0, \tag{2.1a}$$

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = \mathbf{f} - \nabla p + \mu \Delta \mathbf{u}, \tag{2.1b}$$

$$\mathbf{f}(\mathbf{x}, t) = \int_{\Gamma} \mathbf{F}(s, t) \delta(\mathbf{x} - \mathbf{X}(s, t)) \, \mathrm{d}s, \tag{2.1c}$$

$$\mathbf{F}(s, t) = A_f \left(\mathbf{X}(s, t), t \right), \tag{2.1d}$$

$$\frac{\partial \mathbf{X}(s, t)}{\partial t} = \mathbf{u} \left(\mathbf{X}(s, t), t \right) = \int_{\Omega} \mathbf{u}(\mathbf{x}, t) \delta(\mathbf{x} - \mathbf{X}(s, t)) \, \mathrm{d}\mathbf{x}, \tag{2.1e}$$

where t is time, $\mathbf{x}$ the location, $\mathbf{u}$ the velocity field, p the pressure, $\mu = 1/\mathrm{Re}$ the kinematic viscosity, and Re the Reynolds number. The IB Γ is parameterized by s and $\mathbf{X}(s, t)$'s are the Lagrangian points that discretize Γ. $\mathbf{f}(\mathbf{x}, t)$ is the body force exerted on the fluid while $\mathbf{F}(s, t)$ is the internal IB force along Γ. The force generator A_f represents the constitutive law of Γ and its form depends on specific applications. Three most common types are tethering, stretching, and bending. In the case of stretching, we assume that the IB Γ behaves like an elastic fiber [21] so that

$$\mathbf{F}(s, t) = \frac{\partial}{\partial s} \Big(T(s, t) \tau(s, t) \Big), \tag{2.2}$$

where the tangent vector τ is

$$\tau(s, t) = \frac{\partial \mathbf{X}}{\partial s} \Big/ \left\| \frac{\partial \mathbf{X}}{\partial s} \right\|, \tag{2.3}$$

and the tension $T(s, t)$ obeys the Hooke's law

$$T(s, t) = \gamma \left(\left\| \frac{\partial \mathbf{X}}{\partial s} \right\| - L_0 \right). \tag{2.4}$$

Throughout this work it is assumed that the resting length $L_0 = 0$ so that $\mathbf{F}(s, t) = \gamma \frac{\partial^2 \mathbf{X}}{\partial s^2}$, i.e. the force generator is time-independent and has the linear form

$$A_f = \gamma \frac{\partial^2}{\partial s^2}, \tag{2.5}$$

where γ is the stiffness coefficient.

For (2.1) to be well-posed, the initial condition of Γ, the initial condition and boundary conditions for $\mathbf{u}$ are needed. The initial condition and the boundary conditions for pressure are unnecessary up to an additive constant. Throughout this work, we assume periodic boundary conditions. Note that this is not restrictive for practical applications as a wall boundary can be simulated by adding to a periodic domain an additional line of IB points.

The IB method couples the Eulerian grid and the Lagrangian grid through the following spreading and interpolation operators

$$S(\mathbf{F}(s,t)) = \int_\Gamma \mathbf{F}(s,t)\delta(\mathbf{x}-\mathbf{X}(s,t))\,\mathrm{d}s, \tag{2.6}$$

$$S^*(\mathbf{u}(\mathbf{x},\mathbf{t})) = \int_\Omega \mathbf{u}(\mathbf{x},t)\delta(\mathbf{x}-\mathbf{X}(s,t))\,\mathrm{d}\mathbf{x}. \tag{2.7}$$

It follows that (2.1c) and (2.1e) can be rewritten as

$$\mathbf{f}(\mathbf{x},t) = S\left(A_f\mathbf{X}\right), \tag{2.8}$$

$$\frac{\partial \mathbf{X}(s,t)}{\partial t} = S^*\left(\mathbf{u}\left(\mathbf{x},t\right)\right). \tag{2.9}$$

The two operators are adjoint in the sense that

$$\langle S(\mathbf{F}(s,t)), \mathbf{u}(\mathbf{x},t)\rangle_\Omega = \langle \mathbf{F}(s,t), S^*(\mathbf{u}(\mathbf{x},t))\rangle_\Gamma,$$

where the inner products are defined on $L^2(\Omega)$ and $L^2(\Gamma)$, respectively. The total energy of the system can then be expressed as

$$E[\mathbf{u},\mathbf{X}] = \langle \mathbf{u},\mathbf{u}\rangle_\Omega + \langle -A_f\mathbf{X},\mathbf{X}\rangle_\Gamma. \tag{2.10}$$

2.1. The adverse effect of stiffness on stability

An explicit temporal discretization of (2.1) without the convection term yields

$$\nabla\cdot\mathbf{u}^{n+1} = 0, \tag{2.11a}$$

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^{\mathbf{n}}}{\Delta t} = S_nA_f\mathbf{X}^n - \nabla p + \mu\Delta\mathbf{u}^{n+1}, \tag{2.11b}$$

$$\mathbf{X}^{n+1} = \mathbf{X}^n + \Delta t S_n^*\mathbf{u}^{n+1}. \tag{2.11c}$$

Here the word "explicit" refers to the treatment of the body force $\mathbf{f} = SA_f\mathbf{X}$, despite the fact that the diffusion term is treated implicitly in time.

Following Newren et al. [17], we have

$$\begin{aligned}\left\langle 2\mathbf{u}^{n+1}, \mathbf{u}^{n+1}-\mathbf{u}^n\right\rangle_\Omega &= 2\Delta t\left\langle \mathbf{u}^{n+1}, \mu\Delta\mathbf{u}^{n+1} - \nabla p + S_nA_f\mathbf{X}^n\right\rangle_\Omega \\ &= 2\Delta t\mu\left\langle \mathbf{u}^{n+1}, \Delta\mathbf{u}^{n+1}\right\rangle_\Omega + 2\Delta t\left\langle \mathbf{u}^{n+1}, S_nA_f\mathbf{X}^n\right\rangle_\Omega,\end{aligned} \tag{2.12}$$

where we have used the fact that a scalar ψ and a divergence-free vector $\mathbf{u}$ on periodic domains satisfy

$$\int_\Omega \nabla\cdot(\psi\mathbf{u}) = 0 \;\Rightarrow\; \langle \mathbf{u}, \nabla\psi\rangle_\Omega = 0.$$

Also,

$$\begin{aligned}
&\left\langle \mathbf{X}^{n+1}-\mathbf{X}^n, -A_f(\mathbf{X}^{n+1}+\mathbf{X}^n)\right\rangle_\Gamma \\
=&\Delta t\left\langle S_n^*\mathbf{u}^{n+1}, -A_f(2\mathbf{X}^n+\Delta t S_n^*\mathbf{u}^{n+1})\right\rangle_\Gamma \\
=&2\Delta t\left\langle S_n^*\mathbf{u}^{n+1}, -A_f\mathbf{X}^n\right\rangle_\Gamma + \Delta t^2\left\langle S_n^*\mathbf{u}^{n+1}, -A_f S_n^*\mathbf{u}^{n+1})\right\rangle_\Gamma .
\end{aligned} \tag{2.13}$$

Hence the increase of the total energy within one time step for (2.11) is

$$\begin{aligned}
&E[\mathbf{u}^{n+1},\mathbf{X}^{n+1}] - E[\mathbf{u}^n,\mathbf{X}^n] \\
=&\left\langle \mathbf{u}^{n+1}-\mathbf{u}^n, \mathbf{u}^{n+1}+\mathbf{u}^n\right\rangle_\Omega + \left\langle \mathbf{X}^{n+1}, -A_f\mathbf{X}^{n+1}\right\rangle_\Gamma - \left\langle \mathbf{X}^n, -A_f\mathbf{X}^n\right\rangle_\Gamma \\
=&\left\langle \mathbf{u}^{n+1}-\mathbf{u}^n, -\mathbf{u}^{n+1}+\mathbf{u}^n\right\rangle_\Omega + \left\langle \mathbf{u}^{n+1}-\mathbf{u}^n, 2\mathbf{u}^{n+1}\right\rangle_\Omega \\
&+\left\langle \mathbf{X}^{n+1}-\mathbf{X}^n, -A_f(\mathbf{X}^{n+1}+\mathbf{X}^n)\right\rangle_\Gamma \\
=&-\left\langle \mathbf{u}^{n+1}-\mathbf{u}^n, \mathbf{u}^{n+1}-\mathbf{u}^n\right\rangle_\Omega + 2\Delta t\mu\left\langle \mathbf{u}^{n+1}, \Delta\mathbf{u}^{n+1}\right\rangle_\Omega \\
&+\Delta t^2\left\langle S_n^*\mathbf{u}^{n+1}, -A_f S_n^*\mathbf{u}^{n+1}\right\rangle_\Gamma ,
\end{aligned} \tag{2.14}$$

where we have applied the adjointness of S and S^*,

$$2\Delta t\left\langle \mathbf{u}^{n+1}, S_n A_f\mathbf{X}^n\right\rangle_\Omega + 2\Delta t\left\langle S_n^*\mathbf{u}^{n+1}, -A_f\mathbf{X}^n\right\rangle_\Gamma = 0. \tag{2.15}$$

The first two terms in (2.14) are negative-definite. Eq. (2.5) implies the negative-definiteness of A_f, hence the last term is positive-definite. Asymptotically speaking, the last term is much smaller than the other terms. However, when the immersed boundary is stiff, i.e. γ is large, the magnitude of the last term might dominate the first two terms. Consequently, the increase of the total energy during a time step might be positive. To maintain numerical stability, Δt has to be small enough so that the total energy of the discrete system does not increase over time. This explains the adverse effect of high stiffness on the stability of explicit IB methods.

2.2. An implicit discretization

Applying the backward Euler method to the time integration of both velocity and the IB points, we have the following implicit discretization [17] of (2.1):

$$\nabla\cdot\mathbf{u}^{n+1} = 0, \tag{2.16a}$$

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^{\mathbf{n}}}{\Delta t} = S_n A_f\mathbf{X}^{n+1} - \nabla p + \mu\Delta\mathbf{u}^{n+1}, \tag{2.16b}$$

$$\mathbf{X}^{n+1} = \mathbf{X}^n + \Delta t S_n^*\mathbf{u}^{n+1}. \tag{2.16c}$$

Again, "implicit" refers to the treatment of the position of Lagrangian IB markers, not that of the diffusion. Note that the only difference between (2.11) and (2.16) is the change of $\mathbf{X}^n$ to $\mathbf{X}^{n+1}$ in the momentum equation, which adds a new term to (2.12):

$$\begin{aligned}\left\langle 2\mathbf{u}^{n+1}, \mathbf{u}^{n+1}-\mathbf{u}^n\right\rangle_\Omega =& 2\Delta t\mu \left\langle \mathbf{u}^{n+1}, \Delta \mathbf{u}^{n+1}\right\rangle_\Omega + 2\Delta t \left\langle \mathbf{u}^{n+1}, S_n A_f \mathbf{X}^n\right\rangle_\Omega \\ &+ 2\Delta t^2 \left\langle \mathbf{u}^{n+1},\ S_n A_f S_n^* \mathbf{u}^{n+1}\right\rangle_\Omega . \end{aligned} \tag{2.17}$$

Applying (2.17), (2.15), and (2.13) to (2.10), the energy increase of one time step for (2.16) is then

$$\begin{aligned} & E[\mathbf{u}^{n+1}, \mathbf{X}^{n+1}] - E[\mathbf{u}^n, \mathbf{X}^n] \\ =& \left\langle \mathbf{u}^{n+1}-\mathbf{u}^n, -\mathbf{u}^{n+1}+\mathbf{u}^n\right\rangle_\Omega + \left\langle \mathbf{u}^{n+1}-\mathbf{u}^n, 2\mathbf{u}^{n+1}\right\rangle_\Omega \\ & + \left\langle \mathbf{X}^{n+1}-\mathbf{X}^n, -A_f(\mathbf{X}^{n+1}+\mathbf{X}^n)\right\rangle_\Gamma \\ =& -\left\langle \mathbf{u}^{n+1}-\mathbf{u}^n, \mathbf{u}^{n+1}-\mathbf{u}^n\right\rangle_\Omega + 2\Delta t\mu \left\langle \mathbf{u}^{n+1}, \Delta \mathbf{u}^{n+1}\right\rangle_\Omega \\ & - \Delta t^2 \left\langle S_n^* \mathbf{u}^{n+1}, -A_f S_n^* \mathbf{u}^{n+1}\right\rangle_\Gamma . \end{aligned} \tag{2.18}$$

Note how the last term has become negative-definite due to the change of $\mathbf{X}^n$ to $\mathbf{X}^{n+1}$. Now that all terms in (2.18) are negative-definite, clearly the implicit IB method (2.16) is total energy diminishing. Also, changing S_n and S_n^* to S_{n+1} and S_{n+1}^* does not affect this statement.

Although first-order temporal discretizations are used for the exposition, we emphasize that the analysis in this section generalizes in a straight-forward way to second-order temporal discretizations such as the Crank-Nicolson scheme.

3. Algorithms

As discussed in Section 1, obtaining a discretization is one problem, how to solve the resulting linear system efficiently is another. This section discusses the latter.

3.1. Spatial discretization on staggered grids

A staggered Eulerian grid is used to store discrete variables of the flow phase. Referring to Fig. 1, the discrete divergence $\mathbf{D}$, the discrete gradient $\mathbf{G}$, and the discrete Laplacian $\mathbf{L}$[‡] are defined as

$$(\mathbf{D}\mathbf{u})_{i,j} = \frac{u_{i+\frac{1}{2},j} - u_{i-\frac{1}{2},j}}{h} + \frac{v_{i,j+\frac{1}{2}} - v_{i,j-\frac{1}{2}}}{h}, \tag{3.1}$$

$$(\mathbf{G}p)_{i+\frac{1}{2},j} = \frac{p_{i+1,j}-p_{i,j}}{h}, \qquad (\mathbf{G}p)_{i,j+\frac{1}{2}} = \frac{p_{i,j+1}-p_{i,j}}{h}, \tag{3.2}$$

$$(\mathbf{L}p)_{i,j} = \frac{p_{i+1,j}+p_{i-1,j}-2p_{i,j}}{h^2} + \frac{p_{i,j+1}+p_{i,j-1}-2p_{i,j}}{h^2}, \tag{3.3}$$

[‡]We use boldface fonts for discrete operators acting on the quantities over the Eulerian grid and normal fonts for those over the Lagrangian grid.

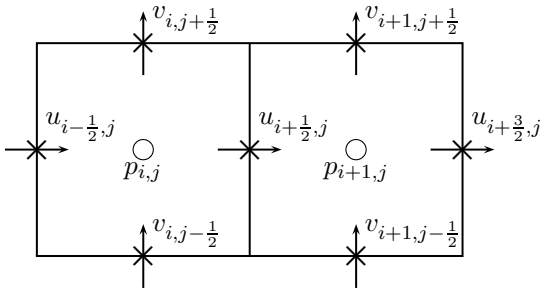


Figure 1: The staggered Eulerian grid. The pressure p is located at the cell centers marked by $\circ$'s while the velocity components u, v the edge centers marked by $\times$'s.

where h is the uniform spacing of the Eulerian grid. It follows that $\mathbf{L} = \mathbf{DG}$.

To numerically implement the spreading and interpolation operators (2.6) and (2.7), we replace the delta function with a regularized discrete delta function presented in [19],

$$\delta_h(\mathbf{x}) = \delta_h(x, y) = \delta_h(x)\delta_h(y), \tag{3.4}$$

$$\delta_h(x) = \begin{cases} \frac{1}{4h}\left(1 + \cos\frac{\pi x}{2h}\right), & |x| \le 2h, \\ 0, & |x| > 2h, \end{cases} \tag{3.5}$$

where $\delta_h(x)$ has a compact support of $4h$, and this localizes the communication between the Eulerian and the Lagrangian grids. As another important feature of $\delta_h(x)$, the Lagrangian IB force is *entirely* transmitted to the Eulerian grid. Eq. (3.4) leads to a discrete version of the spreading operators,

$$S(F^u)_{i+\frac{1}{2},j} = \sum_k F^u_k \delta_h(\mathbf{x}_{i+\frac{1}{2},j} - \mathbf{X}_k)\Delta s, \tag{3.6a}$$

$$S(F^v)_{i,j+\frac{1}{2}} = \sum_k F^v_k \delta_h(\mathbf{x}_{i,j+\frac{1}{2}} - \mathbf{X}_k)\Delta s, \tag{3.6b}$$

and the interpolation operators,

$$S^*(u)_k = \sum_{i,j} u_{i+\frac{1}{2},j}\delta_h(\mathbf{x}_{i+\frac{1}{2},j} - \mathbf{X}_k)h^2, \tag{3.7a}$$

$$S^*(v)_k = \sum_{i,j} v_{i,j+\frac{1}{2}}\delta_h(\mathbf{x}_{i,j+\frac{1}{2}} - \mathbf{X}_k)h^2. \tag{3.7b}$$

Here $\mathbf{F} = (F^u, F^v)$ and $\mathbf{u} = (u, v)^T$. Δs is the spacing of the Lagrangian grids. The subscript k denotes the index of the Lagrangian IB points. With cyclic indexing, the discrete force generator is

$$(A_f\mathbf{X})_k = \frac{\gamma}{\Delta s^2}\left(\mathbf{X}_{k+1} + \mathbf{X}_{k-1} - 2\mathbf{X}_k\right). \tag{3.8}$$

Both (3.6) and (3.7) can be expressed as one equation if different components of the velocity vector and the force vector collocate at the cell center. In contrast, on staggered grids the actions of spreading and interpolation are different for the same IB because the components of a vector to be spreaded and interpolated are at different locations. Consequently, the corresponding matrices for the same operator are different for the vector components.

Despite the additional complexity of the discrete spreading and interpolation operators, we still strongly favor the staggered grid over the collocation grid because the projection operator as in (3.10) satisfies the idempotent condition. Consequently, volume conservation on staggered grids is much better than that on cell-centered grids; this is confirmed by our numerical results.

3.2. A projection solver for the explicit discretization

In 1968, Chorin [6] introduced the projection method with first-order accuracy for the incompressible Navier-Stokes equations. Since then, projection methods have been widely used in computational science and engineering. Meanwhile many variants with second-order accuracy have been developed; some successful examples are those of Kim and Moin [14], Bell et al. [1], and E and Liu [7]. The first author of this paper also proposed a fourth-order approximate projection methods on locally-refined periodic domains [25].

Chorin's projection method first computes an auxiliary velocity field $\mathbf{u}^*$ from the momentum equation by ignoring the pressure gradient term and then project $\mathbf{u}^*$ onto the divergence-free space to fulfill the incompressibility constraint. Apply this approach to the explicit temporal discretization (2.11) and we have

$$\frac{\mathbf{u}^* - \mathbf{u}^{\mathbf{n}}}{\Delta t} = \mu \mathbf{L}\mathbf{u}^* + S_n A_f \mathbf{X}^n, \tag{3.9a}$$

$$\mathbf{u}^{n+1} = \mathbf{P}\mathbf{u}^*, \tag{3.9b}$$

$$\mathbf{X}^{n+1} = \mathbf{X}^n + \Delta t S_n^* \mathbf{u}^{n+1}, \tag{3.9c}$$

where the discrete projection operator

$$\mathbf{P} = \mathbf{I} - \mathbf{G}\mathbf{L}^{-1}\mathbf{D} \tag{3.10}$$

extracts the solenoidal component $\mathbf{u}^{n+1}$ from the vector field $\mathbf{u}^*$. On staggered periodic grids, the discrete operators defined in the previous section satisfy $\mathbf{D} = -\mathbf{G}^T$, $\mathbf{L} = \mathbf{D}\mathbf{G}$, so that $\mathbf{P}$ is idempotent, i.e. $\mathbf{P}^2 = \mathbf{P}$. By (3.9) and (3.10), the divergence-free velocity $\mathbf{u}^{n+1}$ can be obtained by first solving an elliptic system for a scalar field ϕ and then subtracting from the intermediate velocity $\mathbf{u}^*$ a scaled gradient of ϕ:

$$\mathbf{L}\phi = \frac{1}{\Delta t}\mathbf{D}\mathbf{u}^*, \tag{3.11a}$$

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \Delta t \mathbf{G}\phi; \tag{3.11b}$$

hence the projection method can be regarded as a fractional-stepping method.

3.3. Projection method preconditioning

Projection methods provide a means of decoupling the equations for the velocity and the pressure. In most cases, the velocity and pressure from the projection method are not identical to the velocity and pressure that would result from solving the unsplit system. In [9], Griffith showed that projection methods can be very effective preconditioners for solving the unsplit system using Krylov methods. To understand how to use projection methods to precondition the fully coupled system of equations, we begin with the unsplit system of equations for the velocity and pressure:

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^{n}}{\Delta t}+\mathbf{G}p=\mu\mathbf{L}\mathbf{u}^{n+1}+\mathbf{f}, \tag{3.12a}$$

$$\mathbf{D}\mathbf{u}^{n+1}=0. \tag{3.12b}$$

This system of equation can be rearranged and expressed as the block matrix equation

$$\begin{bmatrix}\mathbf{B}_L & \Delta t\mathbf{G}\\ -\Delta t\mathbf{D} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{u}^{n+1}\\ p\end{bmatrix}=\begin{bmatrix}\mathbf{N}\\ \mathbf{0}\end{bmatrix}, \tag{3.13}$$

where

$$\mathbf{B}_L=\mathbf{I}-\mu\Delta t\mathbf{L},\quad\text{and}\quad\mathbf{N}=\mathbf{u}^{n}+\Delta t\mathbf{f}.$$

Note that the divergence constraint has been scaled by $-\Delta t$ to make the matrix symmetric.

Similar to the projection method algorithm, we introduce the change of variables

$$\mathbf{u}^{n+1}=\mathbf{u}^{*}-\Delta t\mathbf{G}\phi, \tag{3.14a}$$

$$p=\mathbf{Y}\phi, \tag{3.14b}$$

where the matrix $\mathbf{Y}$ maps ϕ to the pressure and is to be determined. By substituting this change of variables into (3.13), we arrive at a system of equations for the auxiliary variables

$$\begin{bmatrix}\mathbf{B}_L & -\Delta t\mathbf{B}_L\mathbf{G}+\Delta t\mathbf{G}\mathbf{Y}\\ -\Delta t\mathbf{D} & \Delta t^2\mathbf{D}\mathbf{G}\end{bmatrix}\begin{bmatrix}\mathbf{u}^{*}\\ \phi\end{bmatrix}=\begin{bmatrix}\mathbf{N}\\ \mathbf{0}\end{bmatrix}. \tag{3.15}$$

If it is possible to choose $\mathbf{Y}$ so that

$$-\mathbf{B}_L\mathbf{G}+\mathbf{G}\mathbf{Y}=\mathbf{0}, \tag{3.16}$$

the change of variables results in a block-lower-triangular system

$$\begin{bmatrix}\mathbf{B}_L & \mathbf{0}\\ -\Delta t\mathbf{D} & \Delta t^2\mathbf{D}\mathbf{G}\end{bmatrix}\begin{bmatrix}\mathbf{u}^{*}\\ \phi\end{bmatrix}=\begin{bmatrix}\mathbf{N}\\ \mathbf{0}\end{bmatrix}, \tag{3.17}$$

in which one can solve for $\mathbf{u}^{*}$ and ϕ sequentially. The inverse of this matrix can be expressed as

$$\begin{bmatrix}\mathbf{B}_L & \mathbf{0}\\ -\Delta t\mathbf{D} & \Delta t^2\mathbf{D}\mathbf{G}\end{bmatrix}^{-1}=\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \frac{1}{\Delta t^2}(\mathbf{D}\mathbf{G})^{-1}\end{bmatrix}\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \Delta t\mathbf{D} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{B}_L^{-1} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}. \tag{3.18}$$

Reading these operators from right to left, they correspond to solving for $\mathbf{u}^*$, taking the divergence of $\mathbf{u}^*$, and solving a Poisson equation for ϕ. Therefore, solving this system for the auxiliary variables and then recovering the velocity and pressure using (3.14) is algebraically equivalent to the projection method described in the previous section. Thus, projection methods exactly solve the system (3.17) which approximates the system (3.15). The difference between the solutions of the two systems depends on how closely (3.16) can be satisfied by an appropriate choice of $\mathbf{Y}$.

Perhaps the simplest choice is $\mathbf{Y} = \mathbf{I}$, which corresponds to the pressure update in early projection methods. A better choice for $\mathbf{Y}$ is motivated by Eq. (3.16). If one assumes that $\mathbf{B}_L$ commutes with $\mathbf{G}$, then Eq. (3.16) is satisfied exactly with $\mathbf{Y} = \mathbf{B}_L$[§]. For domains with periodic boundaries $\mathbf{B}_L$ does commute with $\mathbf{G}$ [25, Lemma 4], in which case, a projection method with $\mathbf{Y} = \mathbf{B}_L$ *exactly* solves the unsplit system (3.12).

For other boundary conditions, $\mathbf{B}_L$ commutes with $\mathbf{G}$ at the interior points, but not at points near the boundary. In this case the choice $\mathbf{Y} = \mathbf{B}_L$ only approximately satisfies (3.16), and the projection method alone gives an approximate solution to (3.12). In [9], Griffith showed that a projection method with $\mathbf{Y} = \mathbf{B}_L$ is a very effective preconditioner for solving the unsplit system using a Krylov method for general boundary conditions. Combining (3.18) and (3.14) the projection method preconditioner can be expressed algebraically as

$$\mathbf{P}_{\text{proj}} = \underbrace{\begin{bmatrix} \mathbf{I} & -\Delta t\mathbf{G} \\ \mathbf{0} & \mathbf{Y} \end{bmatrix}}_{\text{transform to original variables}} \underbrace{\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \frac{1}{\Delta t^2}(\mathbf{DG})^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \Delta t\mathbf{D} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{B}_L^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}}_{\text{projection method algorithm}}. \tag{3.19}$$

We adapt this same idea to the implicit-time immersed boundary equations in the next section.

3.4. Implicit-time scheme

We begin with implicit-time scheme from Eq. (2.16) and use Eq. (2.16c) to eliminate $\mathbf{X}^{n+1}$ from Eq. (2.16b) to arrive at the system

$$\nabla \cdot \mathbf{u}^{n+1} = 0, \tag{3.20a}$$

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^{\mathbf{n}}}{\Delta t} = \Delta t S_n A_f S_n^* \mathbf{u}^{n+1} - \nabla p + \mu \Delta \mathbf{u}^{n+1} + S_n A_f \mathbf{X}^n. \tag{3.20b}$$

We replace the differential operators with their discrete counterparts and express the resulting linear system as the matrix equation

$$\mathbf{A} \begin{bmatrix} \mathbf{u}^{n+1} \\ p \end{bmatrix} = \begin{bmatrix} \mathbf{B} & \Delta t\mathbf{G} \\ -\Delta t\mathbf{D} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{u}^{n+1} \\ p \end{bmatrix} = \begin{bmatrix} \mathbf{N} \\ \mathbf{0} \end{bmatrix}, \tag{3.21}$$

[§]We note that this is a slight abuse of notation because $\mathbf{B}_L$ is an operator on vector fields, and $\mathbf{Y}$ is an operator on scalar fields. However, $\mathbf{B}_L$ is block-diagonal with the same operator on the diagonal. One can interpret the scalar version as one of those blocks.

where $\mathbf{A}$ is the saddle point matrix and

$$\mathbf{B} = \mathbf{B}_L - \mathbf{J}, \qquad \mathbf{J} = \Delta t^2 S_n A_f S_n^*. \tag{3.22}$$

This linear system has exactly the same algebraic form as Eq. (3.13), except that the operator $\mathbf{B}$ includes contributions from both the viscous terms and the immersed boundary terms.

The operator $\mathbf{B}$ only differs from $\mathbf{B}_L$ at grid points around the immersed boundary which is a set of codimension one. Thus, it is reasonable to try as a preconditioner

$$\mathbf{P}_{\text{proj}} = \begin{bmatrix} \mathbf{I} & -\Delta t\mathbf{G} \\ \mathbf{0} & \mathbf{Y} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \frac{1}{\Delta t^2}(\mathbf{DG})^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \Delta t\mathbf{D} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{B}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}, \tag{3.23}$$

which is equivalent to (3.19) with $\mathbf{B}$ in place of $\mathbf{B}_L$.

To determine the "ideal" form of $\mathbf{Y}$, left-multiply $\mathbf{P}_{\text{proj}}$ to (3.21), and the matrix on the left side of this equation becomes

$$\mathbf{P}_{\text{proj}} \begin{bmatrix} \mathbf{B} & \Delta t\mathbf{G} \\ -\Delta t\mathbf{D} & \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{V} \\ 0 & \mathbf{W} \end{bmatrix},$$

where $\mathbf{V} = -\Delta t \left\{ \mathbf{B}^{-1}\mathbf{G} + \mathbf{G}\mathbf{L}^{-1}\mathbf{D}\mathbf{B}^{-1}\mathbf{G} \right\}$ and

$$\mathbf{W} = \mathbf{Y}\mathbf{L}^{-1}\mathbf{D}\mathbf{B}^{-1}\mathbf{G}. \tag{3.24}$$

Ideally, the choice of $\mathbf{Y}$ should make $\mathbf{W}$ close to an identity matrix, which can be accomplished with the choice

$$\mathbf{Y} = \left(\mathbf{D}\mathbf{B}^{-1}\mathbf{G}\right)^{-1}\mathbf{D}\mathbf{G}. \tag{3.25}$$

However, the construction of $\mathbf{Y}$ is prohibitively expensive, and the application of $\mathbf{Y}$ involves solving two nontrivial linear systems. We are thus interested in an efficient approximation of $\mathbf{Y}$.

3.4.1. Series approximations

Consider the following series expansion for $\mathbf{B}^{-1}$:

$$\mathbf{B}^{-1} = (\mathbf{B}_L - \mathbf{J})^{-1} = \mathbf{B}_L^{-1}\left(\mathbf{I} - \mathbf{J}\mathbf{B}_L^{-1}\right)^{-1} = \mathbf{B}_L^{-1}\sum_{k=0}^{\infty}\left(\mathbf{J}\mathbf{B}_L^{-1}\right)^k. \tag{3.26}$$

Approximate $\mathbf{B}^{-1}$ with the very first term in the above expansion and we have

$$(\mathbf{B}_L - \mathbf{J})^{-1} \approx \mathbf{B}_L^{-1},$$

which, together with (3.25), yields an approximation to the ideal $\mathbf{Y}$,

$$\mathbf{Y} \approx \mathbf{B}_L, \tag{3.27}$$

where we have used the commutativity of $\mathbf{B}_L$ and $\mathbf{G}$ on periodic domains.

Note that this form of $\mathbf{Y}$ is exactly the same as that used for projection method preconditioning without the immersed boundary. In an effort to include some effect of the immersed boundary, we consider the approximation to $\mathbf{B}^{-1}$ with the first two terms in the expansion (3.26):

$$\mathbf{B}^{-1} = (\mathbf{B}_L - \mathbf{J})^{-1} \approx \mathbf{B}_L^{-1} + \mathbf{B}_L^{-1}\mathbf{J}\mathbf{B}_L^{-1} = (\mathbf{I} + \mathbf{B}_L^{-1}\mathbf{J})\mathbf{B}_L^{-1}.$$

Substituting the above into (3.25)

$$\begin{aligned}\left(\mathbf{D}\mathbf{B}^{-1}\mathbf{G}\right)^{-1}\mathbf{D}\mathbf{G} &\approx \left(\mathbf{D}(\mathbf{I} + \mathbf{B}_L^{-1}\mathbf{J})\mathbf{B}_L^{-1}\mathbf{G}\right)^{-1}\mathbf{D}\mathbf{G} \\ &= \mathbf{B}_L\left(\mathbf{I} + (\mathbf{D}\mathbf{G})^{-1}\mathbf{D}\mathbf{B}_L^{-1}\mathbf{J}\mathbf{G}\right)^{-1} \\ &\approx \mathbf{B}_L\left(\mathbf{I} - (\mathbf{D}\mathbf{G})^{-1}\mathbf{D}\mathbf{B}_L^{-1}\mathbf{J}\mathbf{G}\right) \\ &= \mathbf{B}_L - (\mathbf{D}\mathbf{G})^{-1}\mathbf{D}\mathbf{J}\mathbf{G}.\end{aligned}$$

Our second approximation of the ideal $\mathbf{Y}$ is thus

$$\mathbf{Y} \approx \mathbf{B}_L - (\mathbf{D}\mathbf{G})^{-1}\mathbf{D}\mathbf{J}\mathbf{G}. \tag{3.28}$$

In the above manipulations, we used the fact that $\mathbf{B}_L$ commutes with $\mathbf{D}$ and $\mathbf{G}$, and we assumed that $\left\|(\mathbf{D}\mathbf{G})^{-1}\mathbf{D}\mathbf{B}_L^{-1}\mathbf{J}\mathbf{G}\right\| < 1$. While this last assumption may not be valid for large elastic stiffness, we are seeking an approximation to produce an effective preconditioner which does not require convergence of these series.

3.4.2. Diagonal approximations

We introduce two more choices for $\mathbf{Y}$ that include the immersed boundary by introducing an scalar version of the operator $\mathbf{B}$. The operator $\mathbf{B}$ acts on vector fields while the operator $\mathbf{Y}$ acts on scalar fields. We will define $\mathbf{B}_S = (\mathbf{B}_u + \mathbf{B}_v)/2$ as scalar version of $\mathbf{B}$, where $\mathbf{B}_u$ and $\mathbf{B}_v$ are the diagonal blocks of $\mathbf{B}$. By assuming that

$$\mathbf{B}^{-1}\mathbf{G} \approx \mathbf{G}\mathbf{B}_S^{-1}, \tag{3.29}$$

Eq. (3.25) generates our approximation

$$\mathbf{Y} \approx \mathbf{B}_S. \tag{3.30}$$

Finally, we approximate $\mathbf{B}^{-1}$ in Eq. (3.25) by $\hat{\mathbf{B}}_S^{-1}$ where $\hat{\mathbf{B}}_S = \text{diag}(\mathbf{B}_S)$. Our final approximation to $\mathbf{Y}$ to explore is

$$\mathbf{Y} \approx \left(\mathbf{D}\hat{\mathbf{B}}_S^{-1}\mathbf{G}\right)^{-1}\mathbf{D}\mathbf{G}. \tag{3.31}$$

3.4.3. Summary

Above we have presented a formula for the "ideal" form of pressure update $\mathbf{Y}$ for preconditioning and four different approximations to it given by equations (3.27), (3.28), (3.30), and (3.31). In the next section we test these five different choices for $\mathbf{Y}$ along with $\mathbf{Y} = \mathbf{I}$, because this is sometimes used in projection methods. In summary, the choices of $\mathbf{Y}$ we test are

$\mathbf{Y}(1)$: $\mathbf{Y} = \left(\mathbf{D}\mathbf{B}^{-1}\mathbf{G}\right)^{-1}\mathbf{D}\mathbf{G}$,

$\mathbf{Y}(2)$: $\mathbf{Y} = \mathbf{B}_L = \mathbf{I} - \mu\Delta t\mathbf{D}\mathbf{G}$,

$\mathbf{Y}(3)$: $\mathbf{Y} = \mathbf{B}_L - (\mathbf{D}\mathbf{G})^{-1}\mathbf{D}\mathbf{J}\mathbf{G}$,

$\mathbf{Y}(4)$: $\mathbf{Y} = \mathbf{B}_S$,

$\mathbf{Y}(5)$: $\mathbf{Y} = \left(\mathbf{D}\hat{\mathbf{B}}_S^{-1}\mathbf{G}\right)^{-1}\mathbf{D}\mathbf{G}$,

$\mathbf{Y}(6)$: $\mathbf{Y} = \mathbf{I}$.

To solve the fully coupled system (3.21) we use GMRES with right preconditioning with the projection method preconditioner defined by (3.23). The relative convergence criteria is set to 10^{-8}. Note that the application of the preconditioner involves inverting the matrix $\mathbf{B} = \mathbf{I} - \mu\Delta t\mathbf{L} - \Delta t^2 S_n A_f S_n^*$. For this step we use the multigrid method we developed in [11]. The only difference is that the smoother in this work uses block relaxation instead of point relaxation for a better efficiency of damping high-frequency modes at the IB [12]. Each block contains 5×5 control volumes with an overlap of one control volume in each direction. Within one full multigrid cycle, one presmooth and one postsmooth are performed. Unlike the approach in [12], the smoothing is applied to pressure and different components of the velocity field *separately*. Due to the staggered grids, the forms of the prolongation operator and the restriction operator differ for pressure and each velocity components [23]. For efficiency we approximate the application of $\mathbf{B}^{-1}$ and $(\mathbf{D}\mathbf{G})^{-1}$ by a single F-cycle of multigrid, which is similar to the approach in [9].

4. Tests

A commonly used test problem for immersed boundary methods involves a circular membrane under tension, which is initially stretched in one direction and is then allowed to relax [4, 13, 15, 16, 22, 24]. In this problem, stiffer structures result in faster fluid velocities, and hence the physical time scale is set by the choice of elastic stiffness. However, for efficiency comparison of the explicit method to the proposed method, it is more suitable to use a test problem in which the physical time scale is set by the background flow, not by the stiffness of the structure. In other words, the time step sizes of the test problem should be *solely* constrained by the stiffness of the immersed structure.

To this end, we choose to simulate the forced double-gyre test [10] with the convection term dropped from the governing equations. We nonetheless use Courant number as the measure of time-step sizes so that our results are applicable to the case with the convection terms included. We also restrict Courant numbers to be less than one in the tests since a typical explicit-time discretization of the convection terms would have the same restriction on the time-step sizes.

Out of the different choices of $\mathbf{Y}$, it is found that $\mathbf{Y}(2) = \mathbf{B}_L$ is the best choice for constructing the projection preconditioner.

4.1. Forced double-gyre flow

Consider the double-gyre flow field given by

$$\mathbf{u}_{ss} = \begin{pmatrix} +\sin(2\pi x)\cos(2\pi y) \\ -\cos(2\pi x)\sin(2\pi y) \end{pmatrix}. \tag{4.1}$$

Following [10], an additional forcing of the background flow is added to the RHS of (2.1b) as

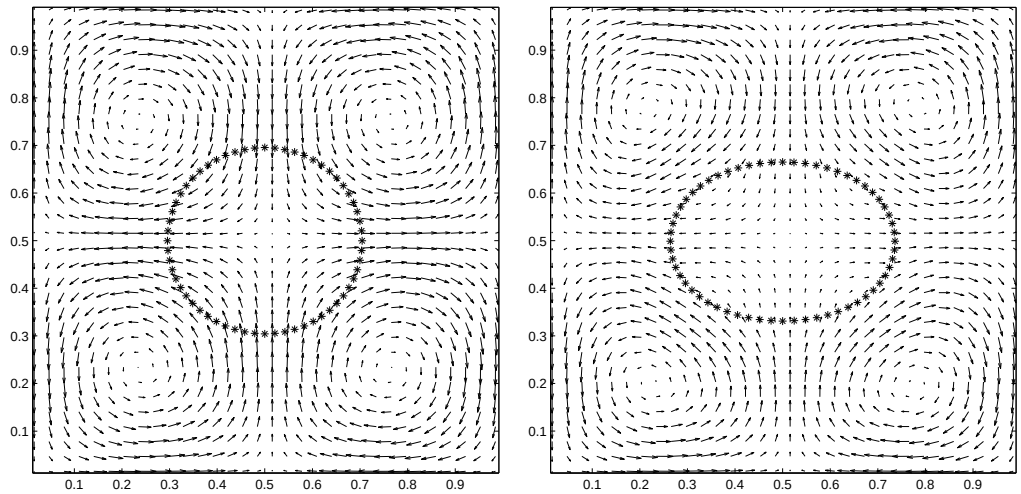
$$\mathbf{f}_{bg} = -(1 - e^{-t})\mu\Delta\mathbf{u}_{ss}, \tag{4.2}$$

which is chosen to drive the steady flow of $\mathbf{u}_{ss}$ as $t \to \infty$ in the absence of the immersed boundary.

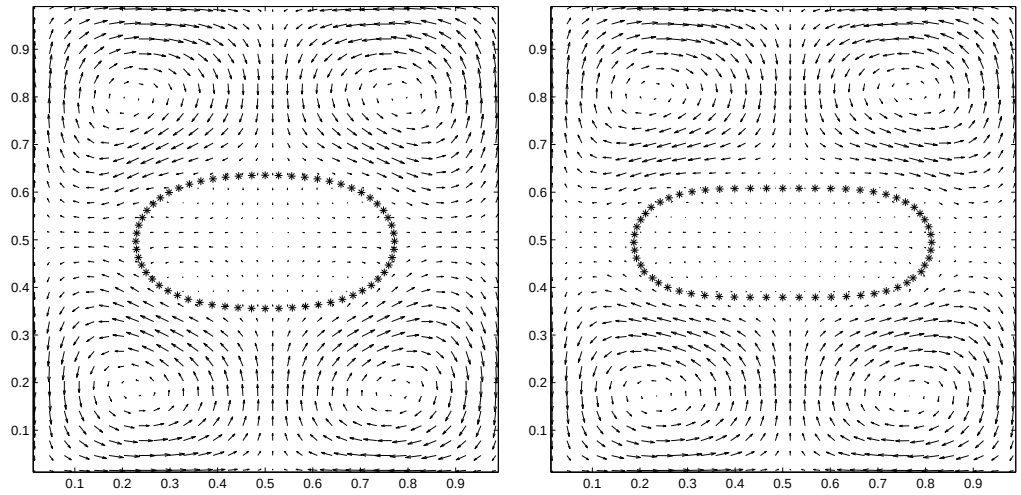
We use the implicit IB method proposed in the previous section to solve the saddle point problem (3.21) with the projection preconditioner (3.19) and $\mathbf{Y} = \mathbf{B}_L$ to advance (2.1) from the initial condition $\mathbf{u}(t_0 = 0, \mathbf{x}) = 0$ to $t_e = 2$ on a unit *periodic* domain $[0, 1] \times [0, 1]$. Fig. 2 illustrates the interaction between the IB and the background flow. To verify the correctness of the program, a refinement study is carried out on five successively refined grids and the results are shown in Table 1, where the first-order convergence on velocity is clearly demonstrated; note that the L_1 and L_2 convergence rates of velocity are limited to one by the temporal discretization of backward Euler. Furthermore, Table 1 demonstrates first-order convergence rates of IB points in 1-norm and confirms the known fact [5] that the pressure computed by IB methods has $\mathcal{O}(1)$ error in max-norm, is first-order accurate in 1-norm, and is half-order accurate in 2-norm. We also note that the explicit IB method discussed in Section 3.2 is also used to perform the same tests, and its results differ from those in Table 1 for only a fraction of one percent.

The results shown in Table 1 and Fig. 2 are obtained using the proposed method with $\mathbf{Y} = \mathbf{B}_L$ as the choice of the projection preconditioner. For the same test problem, we also compare the effectiveness of the six choices of $\mathbf{Y}$. The numbers of iterations of GMRES with different choices of $\mathbf{Y}$ are shown in Fig. 3, from which we make the following observations.

- The iteration number for Y(1) (the ideal choice of $\mathbf{Y}$) is equal or less than 2 for all cases. This confirms the validity of the derivation in Section 3.4. and verifies the correctness of the program.



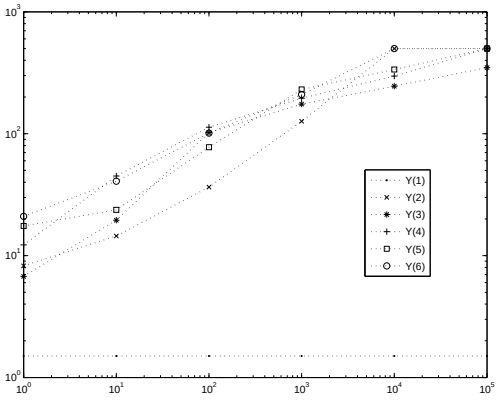
(a) $t = \frac{1}{8}$ (b) $t = \frac{1}{2}$



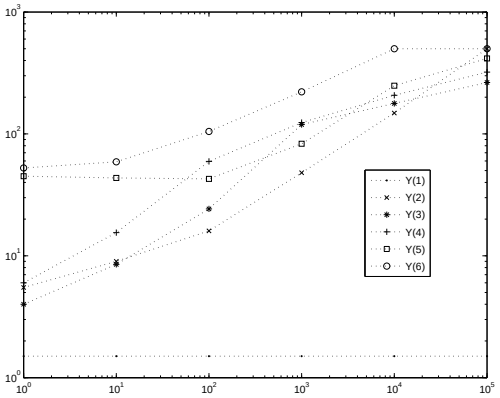
(c) $t = 1$ (d) $t = 2$

Figure 2: Snapshots of the velocity field and IB points produced by the proposed method for the forced double-gyre test on a 32-by-32 grid. $\gamma = 5$, Re$= 1$. '*'s represent the IB Lagrangian points and the arrows the velocity field.

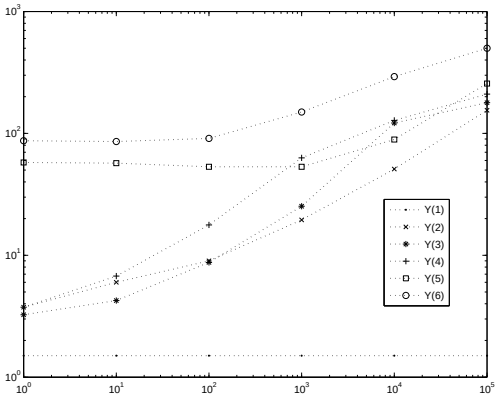
- Aside form Y(1), the best choice of $\mathbf{Y}$ appears to be either Y(2) or Y(3), which are based on the series expansion.
- The iteration numbers for Y(2) and Y(3) decreases as Reynolds number decreases. Indeed, $\left\|\mathbf{J}\mathbf{B}_L^{-1}\right\|$ decreases as μ increases, and consequently $\mathbf{B}^{-1}$ in (3.26) are better approximated by taking the first term or the first two terms in the series expansion.
- The simple pressure update of the identity is almost always the worst choice.



(a) Re=10



(b) Re=1



(c) Re=0.1

Figure 3: The effectiveness of preconditioning for different choices of $\mathbf{Y}$. The abscissa is the stiffness coefficient γ and the ordinate the iteration count of GMRES for solving the saddle point equation (3.21) of the double-gyre problem on a 64^2 grids. Within each iteration, $\mathbf{P}_{\text{proj}}$ in (3.19) is used as the right-preconditioner with a single F-cycle block smoothing for approximating $\mathbf{B}^{-1}$ and $(\mathbf{DG})^{-1}$. The relative convergence criteria is set to 10^{-8}. $Cr = 1$.

Table 1: Errors and convergence rates of the proposed method for the double gyre test. Re$= 1$, $\gamma = 5$, $Cr = 0.5$. Since no analytic solution is available, the convergence rates are calculated by Richardson extrapolation with an error defined as the difference of the results on two adjacent grids. For example, the column titled $\frac{1}{32}-\frac{1}{64}$ represents the difference between the 32^2 grid and the 64^2 grids. To calculate the convergence rate of the IB points on two adjacent grids, we first connect the IB points to form two polygons and then compute the volume of the exclusive disjunction of the two polygons using algorithms from computational geometry. Note that this algorithm for evaluating the errors of IB points is sufficiently accurate because (1) both the expected and the actual convergence rates of IB points are one due to the first-order accuracy of the velocity, (2) the representation of 2D regular semi-analytic sets with polygons is second-order accurate.

h	$\frac{1}{16}-\frac{1}{32}$	rate	$\frac{1}{32}-\frac{1}{64}$	rate	$\frac{1}{64}-\frac{1}{128}$	rate	$\frac{1}{128}-\frac{1}{256}$
u L_1	2.88e-2	1.00	1.44e-2	1.00	7.20e-3	1.00	3.60e-3
u L_2	3.86e-2	1.00	1.93e-2	1.00	9.65e-3	1.00	4.83e-3
u L_∞	1.28e-1	0.73	7.76e-2	0.76	4.58e-2	0.80	2.63e-2
v L_1	2.76e-2	1.12	1.27e-2	1.04	6.18e-3	1.00	3.09e-3
v L_2	3.49e-2	1.14	1.58e-2	1.05	7.63e-3	0.99	3.84e-3
v L_∞	8.54e-2	0.68	5.33e-2	0.75	3.17e-2	0.77	1.86e-2
p L_1	1.27	0.90	0.68	1.05	0.33	0.96	0.17
p L_2	1.91	0.56	1.30	0.70	0.80	0.50	0.56
p L_∞	9.82	-0.08	10.38	0.20	9.04	-0.10	9.68
$\mathbf{X}$ L_1	8.52e-3	0.99	4.28e-3	0.72	2.61e-3	1.05	1.26e-3

4.2. Efficiency of the implicit method

For the same setup with Reynolds numbers Re$= 0.1$, 1, 10, 10^2, 10^3, and stiffness coefficients $\gamma = 1, 10, 10^2, 10^3, 10^4, 10^5$, we list the values of $Cr^{\text{exp}}_{\text{max}}$, the maximum stable Courant numbers, for the explicit IB method in Table 2. For fixed h and Re, $Cr^{\text{exp}}_{\text{max}}$ becomes smaller as γ gets bigger; this confirms the analysis in Section 2.1 that a stiffer IB requires a smaller time step size so as to maintain numerical stability of the explicit IB method. For fixed γ and Re, $Cr^{\text{exp}}_{\text{max}}$ becomes smaller as h is reduced. This is not surprising since a smaller h with the same γ implies a stiffer problem. For a given h, the Reynolds number does not appear to have a substantial influence on $Cr^{\text{exp}}_{\text{max}}$ for stiff IBs. The values of $Cr^{\text{exp}}_{\text{max}}$ for tests different from the forced double-gyre flow are also calculated and found to be qualitatively the same as those in Table 2.

The similarity between (3.13) and (3.21) is amenable to comparing the efficiency of the proposed implicit IB method to that of the explicit IB method. More specifically, we replace $\mathbf{B}^{-1}$ by $\mathbf{B}_L^{-1}$ in (3.19), choose $\mathbf{Y} = \mathbf{B}_L$, and use the corresponding $\mathbf{P}_{\text{proj}}$ to precondition (3.13) before solving it by GMRES. Because multigrid is a very efficient solver for this problem, it makes an excellent preconditioner, and GMRES always converges within two or three iterations [8]. This formulation of the explicit method enables the iteration counts of GMRES to be the single metric for the efficiency comparison as it is independent of machine-specifics such as CPU speed.

In the numerical test, the explicit and implicit IB methods are advanced for n_{test}

Table 2: The maximum Courant numbers for the explicit method (3.13). They are determined by the criterion that (3.13) runs stably for 100 time steps while a 10% increase would yield instability within the same number of time steps. Alternatively, they can be computed via the inequality that maximum eigenvalue of the operator corresponding to the explicit method be less than one. The maximum difference between the results of these two approaches is less than 5%.

		$\gamma = 10^0$	$\gamma = 10^1$	$\gamma = 10^2$	$\gamma = 10^3$	$\gamma = 10^4$	$\gamma = 10^5$
$h = \frac{1}{32}$	Re=0.1	1.00	1.00	6.83e-1	1.12e-1	1.03e-2	1.85e-3
	Re=1	1.00	7.51e-1	1.02e-1	1.83e-2	4.37e-3	1.26e-3
	Re=10	9.09e-1	1.80e-1	4.74e-2	1.37e-2	3.96e-3	1.26e-3
	Re=10^2	4.67e-1	1.35e-1	3.91e-2	1.25e-2	3.96e-3	1.26e-3
	Re=10^3	4.24e-1	1.23e-1	3.91e-2	1.25e-2	3.96e-3	1.26e-3
$h = \frac{1}{64}$	Re=0.1	1.00	1.00	6.83e-1	8.39e-2	9.37e-3	1.53e-3
	Re=1	1.00	7.51e-1	9.23e-2	1.51e-2	3.28e-3	9.51e-4
	Re=10	9.09e-1	1.49e-1	3.56e-2	9.37e-3	2.97e-3	9.47e-4
	Re=10^2	3.51e-1	9.23e-2	2.94e-2	9.37e-3	2.97e-3	8.65e-4
	Re=10^3	2.90e-1	9.23e-2	2.94e-2	9.37e-3	2.97e-3	8.65e-4
$h = \frac{1}{128}$	Re=0.1	1.00	1.00	6.83e-1	8.39e-2	9.37e-3	1.27e-3
	Re=1	1.00	7.51e-1	9.23e-2	1.25e-3	2.71e-3	7.15e-4
	Re=10	8.26e-1	1.23e-1	2.67e-2	7.04e-3	2.04e-3	6.47e-4
	Re=10^2	2.63e-1	6.93e-2	2.01e-2	6.40e-3	2.04e-3	6.47e-4
	Re=10^3	1.98e-1	6.30e-2	2.01e-2	6.40e-3	2.04e-3	6.47e-4

time steps with $Cr = Cr^{\text{exp}}_{\max}$ and $Cr = 1$, respectively. Let N_i^{exp} and N_i^{imp} be the iteration counts of the explicit and implicit IB methods for the ith step, respectively, the speedup of the explicit IB method by the implicit IB method can be measured by the ratio of the averaged iteration count of the former to that of the latter:

$$R_{sp} = \frac{\sum_{i=1}^{n_{\text{test}}} N_i^{\text{exp}}}{Cr^{\text{exp}}_{\max} \sum_{i=1}^{n_{\text{test}}} N_i^{\text{imp}}}. \tag{4.3}$$

The values of R_{sp} for the double-gyre tests are shown in Table 3 for $\mathbf{Y} = \mathbf{B}_L$. When the immersed boundary has low stiffness ($\gamma = 10^0, 10^1, 10^2$), the efficiency advantage of the proposed projection-preconditioning approach is not obvious. In particular, explicit IB method is even faster for $\gamma = 10^0$. However, as the stiffness increases, the efficiency improvement of the proposed method becomes more prominent. We also performed a subset of these test cases on an even finer grid with $h = \frac{1}{256}$. As shown in Table 4, the choice of $\mathbf{Y} = \mathbf{B}_L$ yields a speedup of 29.6 for the highest stiffness of $\gamma = 10^5$ and the lowest Reynolds number Re=0.1.

Besides $\mathbf{Y}(2)$, the only other competitive choice of $\mathbf{Y}$ was $\mathbf{Y}(3)$ (results not shown). For most cases of $\gamma = 10^4, 10^5$ and Re<10, our numerical results show that $\mathbf{Y}(3)$ performs slightly better than $\mathbf{Y}(2)$ for large grid sizes $h = \frac{1}{32}, \frac{1}{64}$, as shown in Fig. 3(a). This is not a surprise. The test cases with large grid sizes correspond to a large ratio of

Table 3: The values of R_{sp}, ratio of the averaged iteration count of the explicit IB method to that of the implicit IB method, for the double-gyre test. $\mathbf{Y} = \mathbf{B}_L$. $n_{\text{test}} = 20$.

		$\gamma = 1$	$\gamma = 10$	$\gamma = 10^2$	$\gamma = 10^3$	$\gamma = 10^4$	$\gamma = 10^5$
$h = \frac{1}{32}$	Re=0.1	0.6	0.3	0.4	1.8	6.8	15.0
	Re=1	0.5	0.5	1.4	4.4	8.1	12.1
	Re=10	0.5	1.5	2.5	2.9	4.0	13.4
	Re=10^2	0.7	1.2	1.4	1.7	4.0	0.7
	Re=10^3	0.7	1.0	1.1	1.2	0.8	0.7
$h = \frac{1}{64}$	Re=0.1	0.7	0.4	0.4	2.3	7.9	18.9
	Re=1	0.6	0.4	2.1	5.4	12.3	12.7
	Re=10	0.5	1.3	4.0	4.9	4.4	4.1
	Re=10^2	1.0	1.7	2.2	1.9	1.4	1.3
	Re=10^3	0.8	1.1	1.3	1.0	1.0	1.1
$h = \frac{1}{128}$	Re=0.1	0.7	0.5	0.4	2.4	8.9	23.3
	Re=1	0.7	0.4	2.2	7.6	16.9	13.7
	Re=10	0.5	1.6	4.3	7.4	5.3	2.2
	Re=10^2	1.0	2.2	3.1	2.1	1.1	0.7
	Re=10^3	1.0	1.2	1.1	0.8	0.7	0.6

Table 4: Results of the double gyre test for $h = \frac{1}{256}$. The test cases here are from a subset of those in Tables 2 and 3.

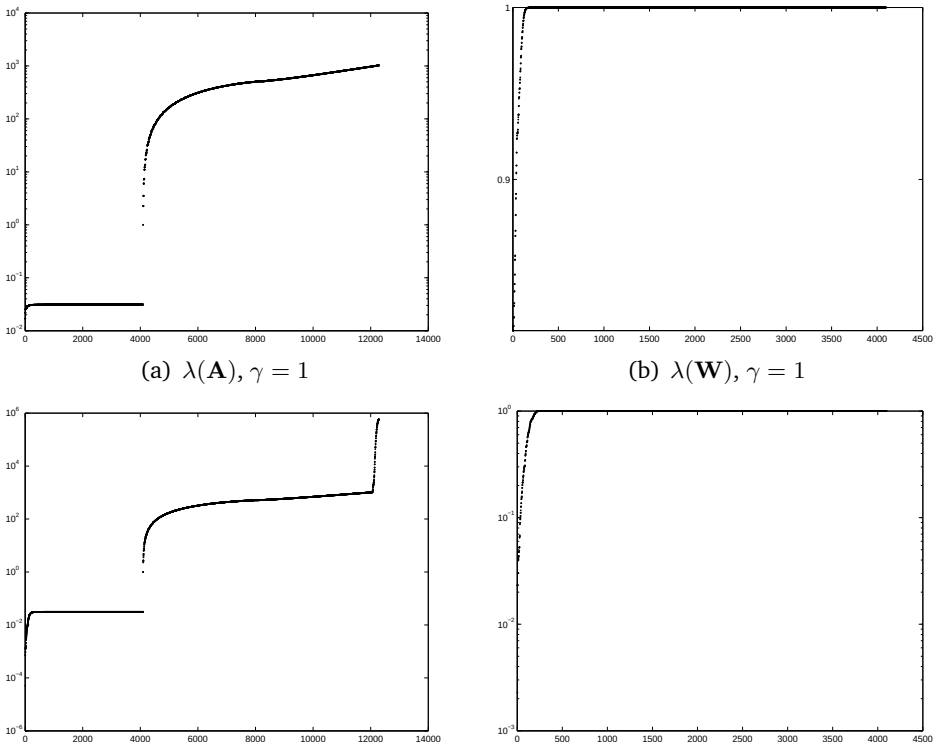
		$\gamma = 10^3$	$\gamma = 10^4$	$\gamma = 10^5$
$Cr_{\max}^{\text{exp}}$	Re=0.1	8.39e-2	9.37e-3	1.04e-3
	Re=1	1.03e-2	2.24e-3	5.91e-4
	Re=10	5.29e-3	1.05e-3	4.02e-4
R_{sp}	Re=0.1	2.4	11.2	29.6
	Re=1	10.2	27.1	17.6
	Re=10	10.5	7.2	1.9

the number of interface cells to that of all cells; thus we expect that a preconditioner that incorporates some effects from the boundary would perform better. However, as the grids are refined, the ratio of the number of interface cells to that of all cells becomes smaller and smaller for a given test. Consequently in our numerical tests, the values of R_{sp} for $\mathbf{Y}(2)$ increases while those for $\mathbf{Y}(3)$ decreases. Considering the additional expense of applying $(\mathbf{DG})^{-1}$ in preconditioning with $\mathbf{Y}(3)$, we prefer to use $\mathbf{Y}(2) = \mathbf{B}_L$ in constructing the projection preconditioner. Nonetheless, the grid size of any numerical simulation is nonzero. If a test has many immersed structures, then a preconditioner that incorporates the effects of the immersed structures may be more effective and worth the extra cost.

The data in Table 3 demonstrate a tendency that a higher stiffness leads to a bigger speedup. In the flow regime of Re$\geq$100, this tendency comes to a stop after the stiffness reaches a certain threshold value. At these threshold values, the speedup drops dramatically; for example, R_{sp} drops from 13.4 to 0.7 from Re$=10$ to Re$=100$ for $h=\frac{1}{32}$ and $\gamma=10^5$. In the flow regime of Re≤ 10, (3.26) and (3.22) suggest that the bigger μ, the better the approximation of $\mathbf{B}_L^{-1}$ to $(\mathbf{B}_L-\mathbf{J})^{-1}$. This is confirmed in Table 3 and Table 4: the projection preconditioner is more effective when the Reynolds number is low.

4.3. Effect of preconditioning on the spectrum

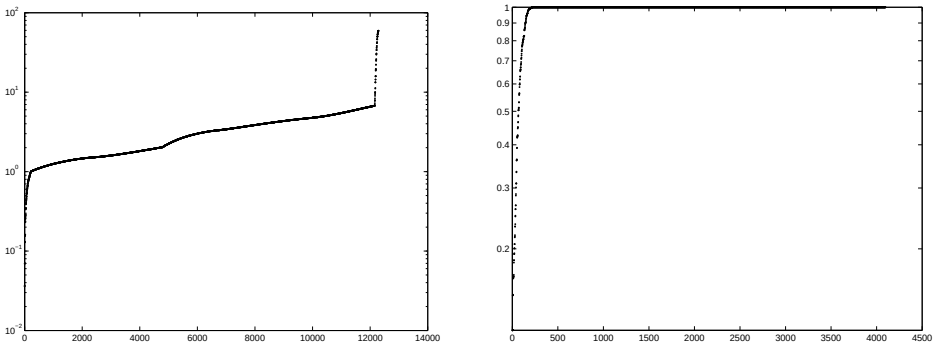
To better understand the results shown in the previous subsection, we plot the spectrum of the saddle point matrix before and after the projection precondition for Re$=1$ in Fig. 4 and for Re$=10^3$ in Fig. 5. As shown in both figures, the eigenvalues



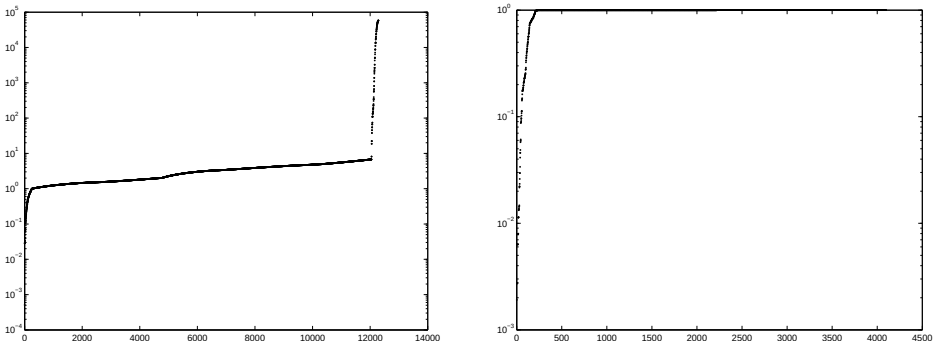
(a) $\lambda(\mathbf{A})$, $\gamma=1$ (b) $\lambda(\mathbf{W})$, $\gamma=1$

(c) $\lambda(\mathbf{A})$, $\gamma=10^4$ (d) $\lambda(\mathbf{W})$, $\gamma=10^4$

Figure 4: Spectrum of the saddle point matrix before ($\mathbf{A}$) and after ($\mathbf{W}$ with $\mathbf{Y}=\mathbf{B}_L$) the projection precondition for Re$=1$ and $h=\frac{1}{64}$ in the double-gyre test at $t=0$. The eigenvalues of each matrix is sorted in magnitude; the x-axis represents the index of an eigenvalue in the sorted array and the y-axis the magnitude of the eigenvalue. The ratio of the number of eigenvalues with their magnitudes less than $1-10^{-6}$ to that of all eigenvalues is 0.0569 for $\gamma=1$ and 0.0600 for $\gamma=10^4$. These ratios are roughly halved when the grid size is reduced to $h=\frac{1}{128}$.



(a) $\lambda(\mathbf{A})$, $\gamma = 1$ (b) $\lambda(\mathbf{W})$, $\gamma = 1$



(c) $\lambda(\mathbf{A})$, $\gamma = 10^4$ (d) $\lambda(\mathbf{W})$, $\gamma = 10^4$

Figure 5: Spectrum of the saddle point matrix before ($\mathbf{A}$) and after ($\mathbf{W}$ with $\mathbf{Y} = \mathbf{B}_L$) the projection precondition for Re$= 10^3$ and $h = \frac{1}{64}$ in the double-gyre test at $t = 0$. The eigenvalues of each matrix is sorted in magnitude; the x-axis represents the index of an eigenvalue in the sorted array and the y-axis the magnitude of the eigenvalue. The ratio of the number of eigenvalues with their magnitudes less than $1 - 10^{-6}$ to that of all eigenvalues is 0.0588 for $\gamma = 1$ and 0.0605 for $\gamma = 10^4$. These ratios are roughly halved when the grid size is reduced to $h = \frac{1}{128}$.

of the original saddle point matrix $\mathbf{A}$ span ten orders of magnitudes for the stiff case $\gamma = 10^4$; this is true in both the low Reynolds number case Re$= 1$ and the high Reynolds number case Re$= 10^3$. For the nonstiff IB $\gamma = 1$, the eigenvalues of the matrix $\mathbf{B}_L$ with high viscosity are bigger than those with low viscosity, hence the low Reynolds number case is more stiff than the high Reynolds number case. Fortunately, the projection preconditioner (3.19) with $\mathbf{Y} = \mathbf{B}_L$ drastically reduces the range of the saddle point matrix to within several orders of magnitudes. As another evidence of the efficiency of the projection preconditioner, the eigenvalues of the preconditioned saddle point matrix $\mathbf{W}$ are highly clustered at one for all the four cases shown in both figures.

The condition numbers of the saddle point matrix before and after preconditioning are shown in Table 5. Clearly, the projection preconditioner based on $\mathbf{Y} = \mathbf{B}_L$ drastically reduces the conditioning of the original saddle point system by orders of magnitudes, especially in the case of low Reynolds numbers. It is important to emphasize that the matrix $\mathbf{B}$ is inverted in computing the eigenvalues of the preconditioned

Table 5: Condition numbers of the saddle point system (3.21) before and after the preconditioning for a 64×64 grid of the double-gyre test ($Cr = 1.0$) at $t = 0$. $C(\mathbf{A})$ is the condition number of $\mathbf{A}$ without preconditioning, $C(\mathbf{W})$ that of $\mathbf{W}$ by computing $\mathbf{B}^{-1}\mathbf{G}$ with Gaussian elimination, and $C(\mathbf{W}_M)$ that of $\mathbf{W}$ by using one F-cycle block iteration to estimate $\mathbf{B}^{-1}\mathbf{G}$. $\mathbf{Y} = \mathbf{B}_L$ for both $C(\mathbf{W})$ and $C(\mathbf{W}_M)$. For low stiffness, the condition numbers quadruple as the grid size is halved. For high stiffness, the condition numbers double as the grid size is halved. Note that the original saddle point matrix $\mathbf{A}$ is symmetric and its condition numbers $C(\mathbf{A})$ corresponds to its spectrums in Figs. 4 and 5. In comparison, the preconditioned matrix $\mathbf{W}$ and its approximation $\mathbf{W}_M$ are not symmetric and their condition number is defined as the ratio of the singular values with the largest and smallest magnitudes.

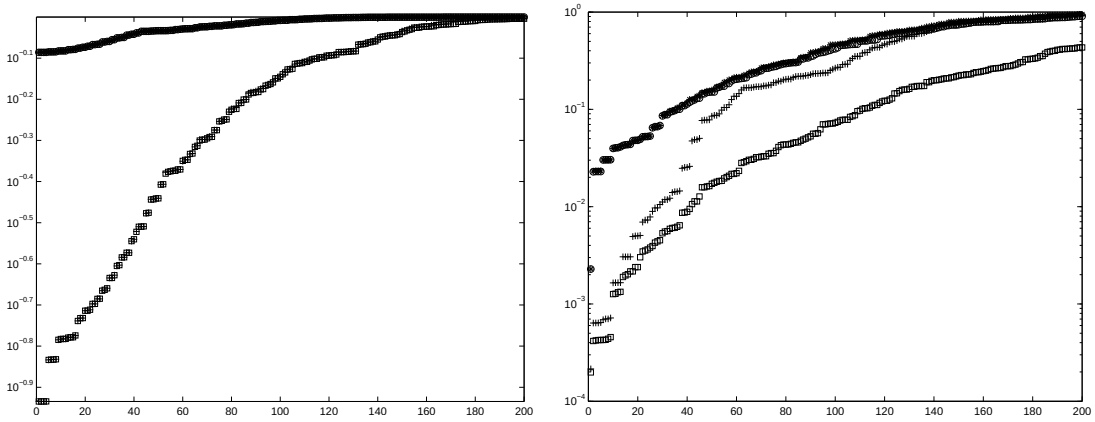
		$\gamma = 1$	$\gamma = 10$	$\gamma = 10^2$	$\gamma = 10^3$	$\gamma = 10^4$	$\gamma = 10^5$
Re=10^3	$C(\mathbf{A})$	1.65e+3	2.31e+4	1.24e+6	1.13e+8	9.98e+9	5.15e+11
	$C(\mathbf{W})$	1.25e+1	7.31e+1	4.34e+2	4.34e+3	3.89e+4	2.03e+05
	$C(\mathbf{W}_M)$	1.26e+1	7.35e+1	4.34e+2	4.40e+3	4.02e+4	2.05e+05
Re=1	$C(\mathbf{A})$	6.08e+4	7.82e+4	1.49e+6	1.18e+8	1.15e+10	1.13e+12
	$C(\mathbf{W})$	1.19e+0	2.38e+0	3.88e+0	4.67e+1	4.57e+02	4.51e+03
	$C(\mathbf{W}_M)$	1.22e+0	2.38e+0	5.54e+0	4.67e+1	4.57e+02	4.52e+03

matrix $\mathbf{W}$ whereas only one full multigrid cycle is applied for those of $\mathbf{W}_M$. In other words, $\lambda(\mathbf{W})$ and $C(\mathbf{W})$ in Figs. 4 and 5 and Table 5 show the *best potential* of the projection preconditioner with $\mathbf{Y} = \mathbf{B}_L$ while $C(\mathbf{W}_M)$ show the *actual performance*. As another notable feature, the reduction of the conditioning is very effective when the inverse of $\mathbf{B}$ is approximated by one full multigrid cycle. These observations not only suggest that the projection preconditioner has a great potential but also imply that this potential can be achieved very efficiently.

To compare the efficiency of the proposed method in different regimes of Reynolds numbers, we first note that the numbers of iterations for the explicit methods for high stiffness are not sensitive to Reynolds numbers: in the case of $\gamma = 10^4$, the iteration number of Re=10^3 is about 10% greater than that of Re=1. Therefore, the results in Table 3 indeed show that the efficiency of the proposed method in the high Reynolds number cases is much worse than that in the low Reynolds number cases. To explain this, the spectrum of $\mathbf{W}$ and $\mathbf{W}_M$ for 200 eigenvalues with the smallest amplitudes are plotted in Fig. 6. For low stiffness, the eigenvalues of $\mathbf{W}$ and $\mathbf{W}_M$ coincide, regardless of the Reynolds numbers. For high stiffness, the eigenvalues of $\mathbf{W}$ and $\mathbf{W}_M$ has a substantial difference for the high Reynolds number case, as shown in Fig. 6(b). In addition, there is only one eigenvalue smaller than 0.01 for Re$= 1$ whereas there are tens of eigenvalues smaller than 0.01 for Re$= 1000$. This might explain the unsatisfactory performance for the proposed implicit IB method in the case of high stiffness and high Reynolds numbers.

5. Conclusion

Based on an unconditionally stable discretization of the IB equations, we have proposed a nonstiff IB method for efficiently simulating fluid-structure interactions at low



(a) $\gamma = 1$ (b) $\gamma = 10^4$

Figure 6: The first 200 eigenvalues with smallest amplitudes for $\mathbf{W}$ and $\mathbf{W}_M$. $h = \frac{1}{64}$. $*$'s and $\circ$'s represent those of $\mathbf{W}$ and $\mathbf{W}_M$ respectively for Re$= 1$; $+$'s and $\square$'s represent those of $\mathbf{W}$ and $\mathbf{W}_M$ respectively for Re$= 10^3$;

Reynolds number. The Lagrangian variables of the IB are eliminated from the governing equations via a Schur complement approach to form purely Eulerian equations, and the corresponding linear system is solved by a Krylov subspace method after being preconditioned by a projection operator.

The projection preconditioner for immersed boundary equations involves approximately inverting an operator for the intermediate velocity and finding an update operator which relates the pressure to the projection variable. We showed that our previously developed multigrid algorithm [11] provides a very good approximate inverse for updating the velocity with just a single multigrid cycle. For the pressure update operator, we used the framework of projection methods to derive an ideal preconditioner, which is essentially the operator resulting from the Schur complement if one were to solve for the pressure first [2]. This operator is very computationally expensive to construct, although we note that in some cases fast algorithms for constructing a similar operator were proposed for implicit immersed boundary equations [4].

We explored a number of choices that approximate the ideal preconditioner. It is found that the proposed method could be thirty times faster than an explicit IB method for low Reynolds number 0.1$\leq$ Re $\leq$10 and high stiffness. In addition, the speedup increases as h and/or the Reynolds number are further reduced. Interestingly, the best choice of pressure update operator is the same as that used in projection preconditioning without the immersed boundary [9]. This update operator ignores the presence of the immersed boundary. While the method worked very well at low Reynolds numbers, the performance suffered above Reynolds number about 100. It may be possible to find a different pressure update operator to provide effective preconditioning at higher Reynolds numbers, but this is beyond the scope of this paper.

Although the values of the speedup for high Reynolds numbers are much less than

those of low Reynolds numbers, this may not be a serious limitation of the proposed method. First, fluid-structure interaction at low Reynolds number covers a vast range of applications, and in the past there have been many applications of the immersed boundary method at low Reynolds numbers. Second, at high Reynolds number it is essential to capture the correct physics of boundary layers. Because of first-order accuracy of the immersed boundary method near the structure, resolving boundary layers requires an extraordinarily large number of grid points. Thus the immersed boundary method is not well suited for applications at very high Reynolds numbers.

Projection methods are appealing because they decouple the pressure and the velocity, and solvers can be built from existing solvers for scalar equations. Another approach is to solve for the pressure and velocity simultaneously. We developed a multigrid-based preconditioner based on this approach for Stokes equation [12], where the time independence precludes the use of projection methods. It remains to be explored whether this approach will be effective at higher Reynolds numbers.

Acknowledgments This work was supported in part by NSF-DMS grants 1160438 and 1226386 to RDG.

References

[1] J. B. Bell, P. Colella, and H. M. Glaz. A second-order projection method for the incompressible Navier-Stokes equations. *J. Comput. Phys.*, 85:257–283, 1989.

[2] M. Benzi, G. H. Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numerica*, pages 1–137, 2005. doi:10.1017/S0962492904000212.

[3] H. D. Ceniceros and J. E. Fisher. A fast, robust, and non-stiff immersed boundary method. *J. Comput. Phys.*, 230:5133–5153, 2011. doi:10.1016/j.jcp.2011.03.037.

[4] H. D. Ceniceros, J. E. Fisher, and A. M. Roma. Efficient solutions to robust, semi-implicit discretizations of the immersed boundary method. *J. Comput. Phys.*, 228(19):7137–7158, 2009.

[5] K.-Y. Chen, K.-A. Feng, Y. Kim, and M.-C. Lai. A note on pressure accuracy in immersed boundary method for Stokes flow. *J. Comput. Phys.*, 230:4377–4383, 2011.

[6] A. J. Chorin. Numerical solution of the Navier-Stokes equations. *Math. Comput.*, 22(104):745–762, 1968.

[7] W. E. and J.-G. Liu. Gauge method for viscous incompressible flows. *Comm. Math. Sci.*, 1(2):317, 2003.

[8] H. C. Elman, D. J. Silvester, and A. J. Wathen. *Finite Elements and Fast Iterative Solvers with Applications in Incompressible Fluid Dynamics*. Oxford University Press, USA, New York, 2005. isbn: 978-0198528685.

[9] B. E. Griffith. An accurate and efficient method for the incompressible Navier-Stokes equations using the projection method as a preconditioner. *J. Comput. Phys.*, 228(20):7565–7595, 2009. doi:10.1016/j.jcp.2009.07.001.

[10] R. D. Guy and A. L. Fogelson. A wave propagation algorithm for viscoelastic fluids with spatially and temporally varying properties. *Comput. Methods Appl. Mech. Engrg.*, 197:2250–2264, 2008. doi:10.1016/j.cma.2007.11.022.

[11] R. D. Guy and B. Philip. A multigrid method for a model of the implicit immersed boundary equations. *Commun. Comput. Phys.*, 12(2):378–400, 2012.
[12] R. D. Guy, B. Philip, and B. E. Griffith. Geometric multigrid for an implicit-time immersed boundary method. *Advances in Computational Mathematics*, in press, 2014. doi:10.1007/s10444-014-9380-1.
[13] T. Y. Hou and Z. Shi. An efficient semi-implicit immersed boundary method for the Navier-Stokes equations. *J. Comput. Phys.*, 227:8968–8991, 2008.
[14] J. Kim and P. Moin. Application of a fractional-step method to incompressible Navier-Stokes equations. *J. Comput. Phys.*, 59(2):308–323, 1985.
[15] A. A. Mayo and C. S. Peskin. An implicit numerical method for fluid dynamics problems with immersed elastic boundaries. In *Fluid dynamics in biology (Seattle, WA, 1991)*, volume 141 of *Contemp. Math.*, pages 261–277. Amer. Math. Soc., Providence, RI, 1993.
[16] Y. Mori and C. S. Peskin. Implicit second-order immersed boundary methods with boundary mass. *Comput. Methods Appl. Mech. Engrg.*, 197(25-28):2049 – 2067, 2008.
[17] E. P. Newren, A. L. Fogelson, R. D. Guy, and R. M. Kirby. Unconditionally stable discretizations of the immersed boundary equations. *J. Comput. Phys.*, 222:702–719, 2007. doi:10.1016/j.jcp.2006.08.004.
[18] E. P. Newren, A. L. Fogelson, R. D. Guy, and R. M. Kirby. A comparison of implicit solvers for the immersed boundary equations. *Comput. Methods Appl. Mech. Engrg.*, 197:2290–2304, 2009. doi:10.1016/j.cma.2007.11.030.
[19] C. S. Peskin. Numerical analysis of blood flow in the heart. *J. Comput. Phys.*, 25:220–252, 1977.
[20] C. S. Peskin. The immersed boundary method. *Acta Numerica*, pages 479–517, 2002. doi:10.1017/S0962492902000077.
[21] C. S. Peskin and D. M. McQueen. A three-dimensional computational method for blood flow in the heart: I. immersed elastic fibers in a viscous incompressible fluid. *J. Comput. Phys.*, 81:372–405, 1989.
[22] J. M. Stockie and B. R. Wetton. Analysis of stiffness in the immersed boundary method and implications for time-stepping schemes. *J. Comput. Phys.*, 154(1):41–64, 1999.
[23] U. Trottenberg, C. Oosterlee, and A. Schuller. *Multigrid*. Academic Press, San Diego, CA, 2001. ISBN:0-12-701070-X.
[24] C. Tu and C. S. Peskin. Stability and instability in the computation of flows with moving immersed boundaries: A comparison of three methods. *SIAM J. Sci. Stat. Comput.*, 13(6):1361–1376, 1992.
[25] Q. Zhang. A fourth-order approximate projection method for the incompressible Navier-Stokes equations on locally-refined periodic domains. *Appl. Numer. Math.*, 77(C):16–30, 2014.